Shared Virtual Memory and Message Passing Programming on a Finite Element Application

Rudolf Berrendorf, Michael Gerndt, Zakaria Lahjomri, Thierry Priol

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Shared Virtual Memory and Message Passing Programming on a Finite Element Application

Rudolf Berrendorf and Michael Gerndt, Zakaria Lahjomri and Thierry Priol

N° 2355

Mars 1995

PROGRAMME 1

ISSN 0249-6399

Shared Virtual Memory and Message Passing Programming on a Finite Element Application *

Rudolf Berrendorf and Michael Gerndt**, Zakaria Lahjomri and Thierry Priol ***

Programme 1 — Architectures parallèles, bases de données, réseaux et systèmes distribués
Projet Caps

Rapport de recherche n° 2355 — Mars 1995 — 14 pages

Abstract: This paper describes the methods used and experiences made with implementing a finite element application on three different parallel computers with either message passing or shared virtual memory as the programming model. Designing a parallel finite element application using message-passing requires to find a data domain decomposition to map data into the local memory of the processors. Since data accesses may be very irregular, communication patterns are unknown prior to the parallel execution and thus makes the parallelization a difficult task. We argue that the use of a shared virtual memory greatly simplifies the parallelization step. It is shown experimentally on an hypercube iPSC/2 that the use of the KOAN/Fortran-S programming environment based on a shared virtual memory allows to port quickly and easily a sequential application without a significant degradation in performance compared to the message passing version. Results for recent parallel architectures such as the Paragon XP/S for message-passing and the KSR1 for shared virtual memory are presented, too.

(Résumé : tsvp)

*This work is supported by the Esprit BRA APPARC
**Zentralinstitut für Angewandte Mathematik, Forschungszentrum Jülich (KFA), D-52425 Jülich - Germany
***IRISA-INRIA

Unité de recherche INRIA Rennes
IRISA, Campus universitaire de Beaulieu, 35042 RENNES Cedex (France)
Téléphone : (33) 99 84 71 00 – Télécopie : (33) 99 84 71 71

Comparaison des techniques de mémoire virtuelle partagée et d'échange de messages à l'aide d'une application par élément finie

Résumé : Ce papier présente les résultats d'une mise en oeuvre d'une application par élément finie sur trois architectures parallèles en utilisant soit l'échange de messages soit une mémoire virtuelle partagée. La conception d'une application par élément finie sur une architecture parallèle à mémoire distribuée nécessite une décomposition de domaine afin de placer les données dans les mémoires locales des processeurs. Les accès aux données étant irréguliers, la parallélisation est une tâche complexe. Nous montrons que l'utilisation d'une mémoire virtuelle partagée simplifie grandement cette tâche. Nous montrons, à l'aide de résultats expérimentaux sur un hypercube iPSC/2, que l'utilisation de l'environnement de programmation KOAN/Fortran-S permet le portage rapide d'une telle application sans constater une perte significative. Nous présentons également des résultats sur des machines plus récentes comme le Paragon XP/S et la KSR-1.

1 Introduction

Parallelizing applications with irregular data access, such as those using finite element techniques, on distributed memory parallel computers is a complex task due the distributed nature of the memory. Porting such applications on distributed memory parallel computers requires to handle arbitrary data distributions as the data accesses are unknown at compile time. The PARTI subroutine package developed at NASA/ICASE by J. Saltz et al. [5] has been designed for that purpose. It allows the computation of processor-local indices, the analysis of communication patterns and the communication of non-local array elements. By using PARTI, few modifications to the sequential application are necessary to get a parallel version. However, the user is still responsible to select those arrays to be distributed, to specify the distributions, and to carefully analyze and transform references to distributed arrays. The specification of the distributions as well as the code analysis requires deeper knowledge of the application and is a time consuming and error-prone task.

To avoid this tedious task, one can use a shared virtual memory (SVM) concept so as to program distributed memory parallel computers like conventional shared memory parallel architectures. A SVM hides the physical local memories and provides to the user a virtual address space made of pages that move on demand among processors. Each local memory acts as large cache. This concept can be implemented within the operating system such as the KOAN SVM [6] or by specialized hardware devices as done in the KSR1 of Kendall Square Research. However, using an SVM will add additional overhead to the parallel execution caused by several factors like a distribution overhead, cache coherency, etc. The aim of this paper is to provide a comparison between the two programming models with respect to programming aspects and performance. A comparing experiment is carried out using the same machine: an hypercube iPSC/2. Since this machine does not represent the state of the art in the design of parallel architectures, we present additionally the results for two recent parallel architectures: the Intel Paragon XP/S and the KSR1.

This paper is organized as follows. Section 2 describes the ParFEM application we used in the test. Section 3 addresses the parallelization of ParFEM using a message-passing programming model whereas section 4 and 5 deals with the shared virtual memory programming model. Section 6 discusses several issues for the two programming models. We conclude in section 7.

2 ParFEM: A Finite Element Application

The ParFEM application has been developed by Harry Vereecken et al. (Institute for Petrol and Organic Geochemistry) [9] at KFA. This application models transport and chemical processes in heterogeneous 3D porous media. Accounting for the heterogeneity of the porous medium results in grid size of more than 10^6 nodal points. In combination with the strong nonlinearity of the partial differential equations such problems can only be handled on high performance computer systems in combination with appropriate numerical solution techniques for the linearized set of equations. These equations are obtained by applying